Fundamentos em Engenharia de Dados

Marcos Claver

2024

SUMÁRIO

Capítulo 1. Conceitos Fundamentais de Engenharia de Dados 5

1.1. Conceitos Fundamentais 5

1.2. O Processo de Engenharia de Dados 5

1.3. Pipeline de Dados 6

1.4. ETL x ELT 8

1.5. Tipos de Workloads de Dados 9

1.6. Processamento de dados (Streaming, Batch, Micro Batch) 11

1.7. Big Data, 5V's? 14

Capítulo 2. File Types e Data Types 17

2.1. Formatos de arquivos mais usados no pipeline de Engenharia de Dados 17

2.2. Data Types 18

Capítulo 3. Arquiteturas 22

3.1. OLTP 22

3.2. OLAP 23

3.3. Lambda 25

3.4. EDA – Event-Driven Architecture 27

Capítulo 4. Técnicas de Coleta de Dados 30

4.1. Crawler 30

4.2. Scrapping 30

4.3. API – Application Programming Interface 31

Capítulo 5. Arquitetura de Microsserviços 34

5.1. Conceitos e Aplicações 34

5.2. Containers 35

5.3. Docker e Kubernetes: conceitos básicos 36

Capítulo 6. Data Governance 40
6.1. Governança de Dados 40
6.2. DAMA 40
6.3. Dama DMBOK 41
Capítulo 7. Mineração de dados 44
7.1. Pré-processamento de dados: limpeza, integração e transformação 45
7.2. Seleção de atributos 46
7.3. Modelos 46
Capítulo 8. Data Governance 50
8.1 Manifesto Ágil 50
8.2. Devops 52
8.3. DataOps 53
8.4. FinOps 58
Capítulo 9. Modern Data Stack 61
9.1. Data Mesh 61
9.2. Amazon Aurora zero-ETL 61
9.3. MWAA – Amazon Managed Workflows for Apache Airflow 62
Capítulo 10. Boas Práticas 65
10.1. Clean Code 65
10.2. Versionamento 65
10.3. Documentação 66
Referências 67

> Capítulo 1

Capítulo 1. Conceitos Fundamentais de Engenharia de Dados

1.1. Conceitos Fundamentais

Desde a criação da internet, passando pela popularização dos computadores bem como o advento da internet, até o “Big Data”, várias fontes de dados foram criadas da mesma forma que outras se tornaram obsoletas, exemplo disso são as Listas Telefônicas e as Enciclopédias. Junto dessa mudança profissões também foram impactadas, algumas deixando de existir, outras se adaptando e novas surgindo, dentre essas o Engenheiro de Dados.

A Engenharia de Dados muitas vezes é confundida com a Engenharia tradicional, como a Engenharia Civil ou Mecânica, para essas é necessária uma formação específica, bem como um Conselho. Já a Engenharia de Dados está mais ligada com habilidades pessoais e técnicas do que com uma formação específica.

Mas o que é a Engenharia de Dados? É um setor dentro da área de dados, responsável por coletar, processar, enriquecer, armazenar e disponibilizar dados – não necessariamente nessa ordem – ou seja, é a área que faz a ponte entre a origem dos dados até o consumo dos dados por Analistas de Dados, Cientistas de Dados e Engenheiros de Machine Learning, para que esses profissionais possam fazer suas análises e estudos extraindo informações para que os Stakeholders possam tomar suas decisões.

1.2. O Processo de Engenharia de Dados

O processo se inicia muitas vezes por uma dor do time de negócios ou por uma oportunidade, que serão atendidas com ajuda de dados.

O time de engenharia recebe a demanda com as informações iniciais, levanta os requisitos técnicos e retorna para o time de negócio para alinhamento e expectativas e caso existam retirar dúvidas.

Algumas perguntas acerca do fluxo de dados devem ser feitas para um desenvolvimento alinhado com o demandante:

- Quem terá acesso aos dados?
- Existem dados pessoais, se sim, podem ser mascarados?
- Qual a frequência de atualização dos dados?
- Qual o método de atualização dos dados?

Esclarecidas as dúvidas, inicia-se etapa de desenvolvimento do pipeline de dados. Segundo a AWS, pipeline de dados consiste em:

Após o desenvolvimento, começa a etapa de homologação, onde o pipeline é executado num ambiente apartado do ambiente e produção, a fim de se obter dados. Então se libera o acesso para os usuários validarem os dados e mantêm-se o pipeline em execução conforme a frequência previamente acordada, para que a atualização das tabelas ocorra, e verifica-se após essas. Caso os dados não estejam corretos, são levantados exemplos de inconsistências e começa uma etapa de revisão para resolver os problemas. Com os dados e o fluxo validados, o pipeline começa a ser executado no ambiente de produção restando somente ao Engenheiro(a) documentar o pipeline de dados.

1.3. Pipeline de Dados

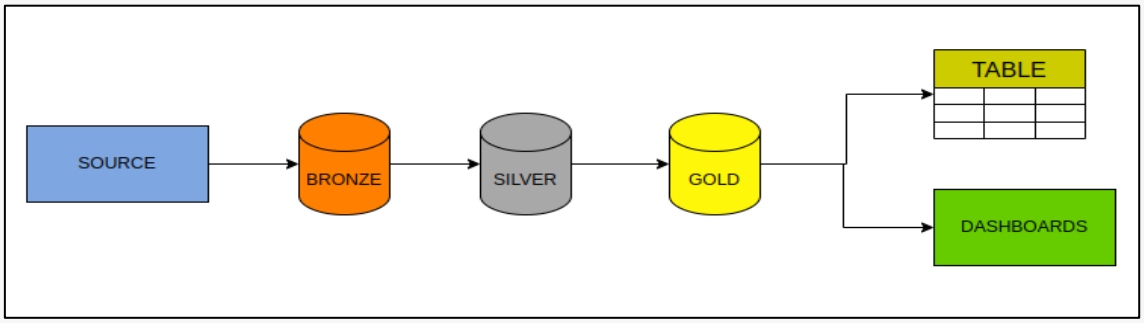
Primeiramente vamos entender o que seria um pipeline. Segundo o dicionário, pipeline significa “canalização, tubulação usada para o transporte a grandes distâncias de fluidos, especialmente petróleo (oleoduto) ou gás (gasoduto)”, dessa forma podemos entender pipeline de dados como um

fluxo de transporte de dados, buscando na origem e levando até o destino. Completando esse entendimento, temos o conceito aplicado pela AWS:

> [...] uma série de etapas de processamento para preparar dados corporativos para análise. As organizações têm um grande volume de dados de várias fontes, como aplicativos, dispositivos de Internet das Coisas (IoT) e outros canais digitais. No entanto, os dados brutos são inúteis; eles devem ser movidos, classificados, filtrados, reformatados e analisados para business intelligence. Um pipeline de dados inclui várias tecnologias para verificar, resumir e encontrar padrões nos dados para informar as decisões de negócios. Pipelines de dados bem-organizados oferecem suporte a vários projetos de big data, como visualizações de dados, análises exploratórias de dados e tarefas de machine learning. (Amazon Web Services, 2023)

A fim de ilustrar, vamos entender o pipeline abaixo representando uma ingestão em um data lake, tema que abordaremos mais à frente:

Figura 1 – Pipeline de Dados.



1. Origem dos dados ou fonte, podendo variar desde um aplicativo, um software de registro de vendas, sensores etc.

2. Camada Bronze, onde os dados gerados na fonte são armazenados sem alterações.

3. Camada Silver, os dados já recebem tratamentos, por exemplo correção ou alteração de datas, limpeza de dados, alteração de

datatypes – numéricos, booleans – não se preocupem, pois falaremos sobre datatypes mais à frente.

4. Camada Gold, os dados tratados agora são enriquecidos, agrupados entre outras atividades que atendam ao objetivo para o qual são necessários, é a camada onde Negócios e Análise de Dados consomem os dados, seja para gerar tabelas ou para gerar dashboards.

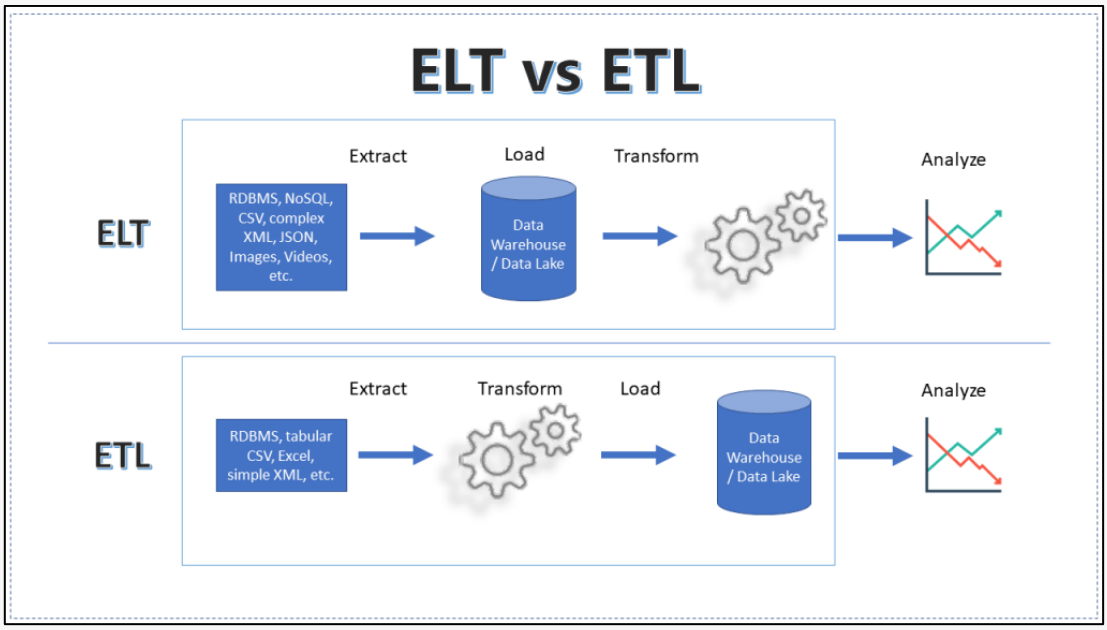
1.4. ETL x ELT

Entendido o conceito de pipeline de dados assim como sua aplicação, vamos entender suas variações.

Extract, Transform and Load (ETL), trazendo para o português brasileiro: Extração, Transformação e Carga. Os dados são extraídos, transformados e em sequência armazenados, diferentemente do exemplo anterior, nesse caso os dados são armazenados somente após serem tratados, muito utilizado para alimentar os famosos “Data Warehousing”.

Extract, Load and Transform, (ELT), no português brasileiro: Extração, Carga e Transformação. Conforme o exemplo da imagem, apresenta as mesmas etapas do item anterior, mas em uma ordem diferente, onde os dados são extraídos, carregados e depois armazenados, comumente visto como ELTL, acrescentando uma etapa de carregamento dos dados após o tratamento. Esse processo é utilizado para pipelines em streaming e/ou micro batch.

Figura 2 – ETL x ELT



Fonte: https://blog.skyvia.com/elt-vs-etl/.

1.5. Tipos de Workloads de Dados

Transacional (OLTP): Online Transactional Processing

São os sistemas transacionais das empresas, utilizado para processar transações que ocorrem simultaneamente nas empresas. Conforme a Oracle comenta:

> No passado, o OLTP era limitado a interações do mundo real em que algo era trocado, como por dinheiro, produtos, informações, solicitação de serviços e assim por diante. Mas a definição de transação nesse contexto se expandiu ao longo dos anos, principalmente desde o advento da internet, abrangendo qualquer tipo de interação digital ou engajamento com uma empresa que possa ser acionado de qualquer lugar do mundo e por meio de qualquer sensor conectado à web Isso também inclui qualquer tipo de interação ou ação, como fazer download de pdfs em uma página da web, visualizar um vídeo específico ou acionadores de manutenção automática ou comentários em canais sociais que podem ser críticos para uma empresa registrar a fim de atender melhor seus clientes. (Oracle, 2023)

Analítico (OLAP): Online Analytics Processing

Um processamento voltado para análise, comumente utilizado para Data Warehouses. Conforme a IBM sugere, as ferramentas OLAP são projetadas para análise multidimensional de dados em um data warehouse, que contém dados tanto transacionais quanto históricos.

OLTP x OLAP

Podemos já visualizar diferenças somente pelo nome, onde um refere-se ao processamento de transações e o outro ao processamento analítico. Mas podemos comparar os sistemas em um diagrama, para facilitar o entendimento:

Tabela 1 – OLTP x OLAP.

Sistemas OLTP	Sistemas OLAP
Habilita a execução em tempo real de um grande número de transações de banco de dados por um grande número de pessoas.	Geralmente envolve a consulta de muitos registros (mesmo todos os registros) em um banco de dados para fins analíticos.
Requer tempos de resposta extremamente rápidos.	Requer tempos de resposta que são de ordens de magnitude mais lentas do que os exigidos pelo OLTP.
Modifica pequenas quantidades de dados com frequência e geralmente envolve um equilíbrio de leituras e gravações.	Não modifica os dados de forma alguma, cargas de trabalho são geralmente de leitura intensiva.
Usa dados indexados para melhorar os tempos de resposta.	Armazena dados em formato colunar para permitir acesso fácil a um grande número de registros.

Exige backups de banco de dados frequentes ou simultâneos.	Exige backups de banco de dados muito menos frequentes.
Exige relativamente pouco espaço de armazenamento.	Normalmente tem requisitos de espaço de armazenamento significativos, porque armazenam grandes quantidades de dados históricos.
Geralmente executa consultas simples envolvendo apenas um ou alguns registros.	Executa consultas complexas envolvendo um grande número de registros.

Fonte: https://www.oracle.com/br/database/what-is-oltp/

Dessa forma, concluímos que OLTP é utilizado para alteração de transações online e o OLAP para análise de dados históricos para recuperar informações.

1.6. Processamento de dados (Streaming, Batch, Micro Batch)

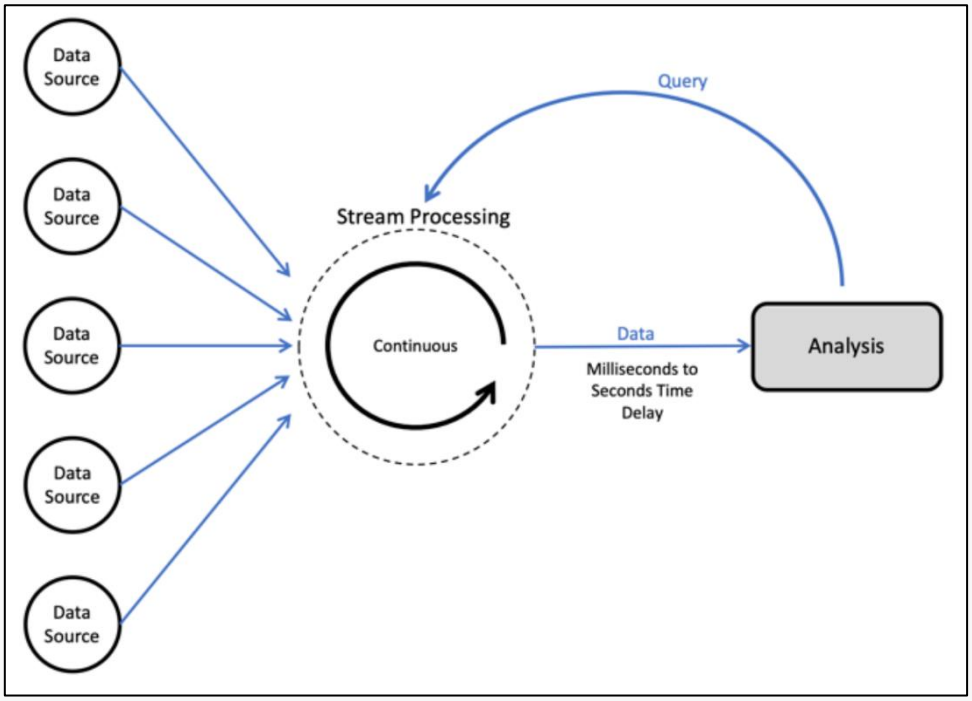
Alguém pode ler Streaming e achar que se trata de plataformas áudio visuais, mas não é isso. Segundo o “Streaming Systems: The What, Where, When, and How of Large-Scale Data Processing”, citado por Vasconcelos Luiz et al., streaming de dados é “um tipo de engine de processamento de dados projetado para tratar datasets infinitos”. Outro ponto importante a se destacar desse tipo de processamento, é que ele trabalha com duas pontas, uma entrada de dados e uma saída de dados, Vasconcelos Luiz et al., faz uma analogia bem interessante a cerca dessa “estrutura”:

> Pensem em uma caixa d’água que é enchida por meio de uma ligação de canos da companhia de água da sua cidade. Essa caixa d’água dá vazão para as torneiras e chuveiros da casa por meio da ligação hidráulica de sua residência. Nessa abstração,

> nosso dataset é a água contida na caixa d'água, nosso source é a ligação com a companhia de água e, finalmente, os sinks são as torneiras e chuveiros. (Vasconcelos et al., 2020)

Abaixo, temos o exemplo de uma arquitetura de um processamento em streaming.

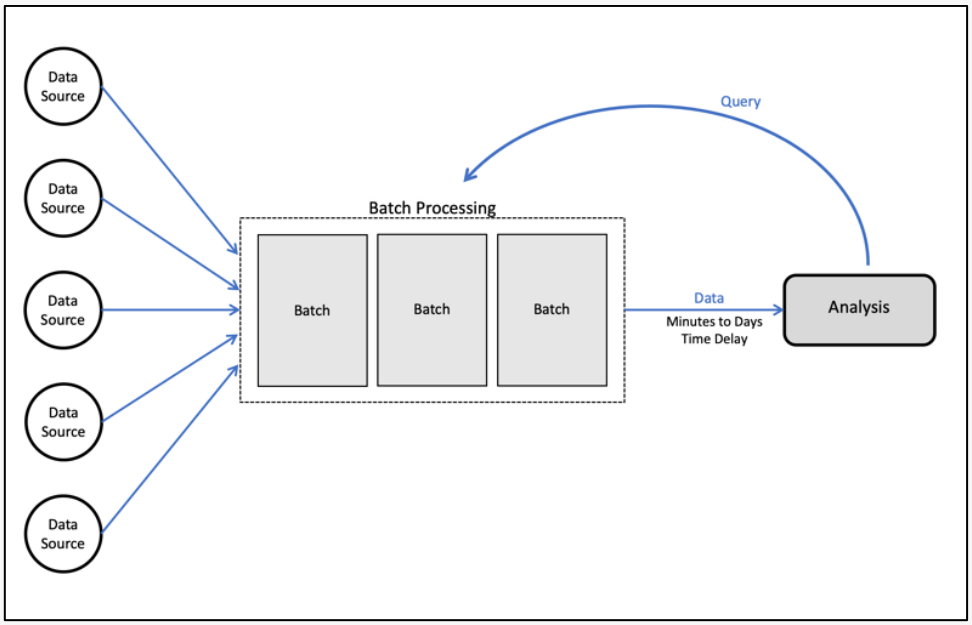
Figura 3 – Streaming Processing.



Fonte: https://www.upsolver.com/blog/batch-stream-a-cheat-sheet.

Batch, traduzindo para o português, significa lote, ou seja, o processamento acontece em lotes de dados, grupo de dados e não de forma contínua como em um processamento em streaming, como num exemplo citado anteriormente das transações bancárias, os dados das transações estão em um lote, no caso arquivo, onde o pipeline que processará o arquivo executará somente uma vez.

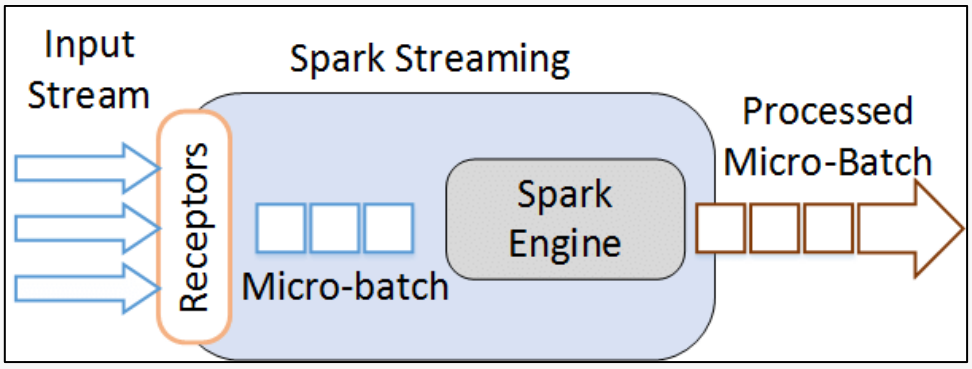
Figura 4 – Batch Processing.



Fonte: https://www.upsolver.com/blog/batch-stream-a-cheat-sheet.

Já o Micro Batch seria o intermediário entre o Streaming e o Batch, onde ocorre processamento de dados em pequenos lotes, sendo possível processar grandes datasets com uma latência baixa. Segundo Eran Levy, no seu artigo “Batch vs Stream vs Microbatch Processing: A Cheat Sheet”, micro batch pode ser entendido como um método de processamento eficiente de grandes conjuntos de dados com latência reduzida e escalabilidade aprimorada. Ele divide grandes conjuntos de dados em lotes menores e os executa em paralelo, resultando em um processamento mais oportuno e preciso.

Figura 5 – Microbatch Processing.



Fonte: https://www.researchgate.net/figure/Micro-batch-processing-used-in-Spark-stream-The-input-streams-are-received-by_fig6_327183263.

O tipo de processamento irá variar com o tipo de demanda, como os dados são gerados e qual o tamanho deles. Não há um tipo melhor que o outro, são para necessidades diferentes.

1.7. Big Data, 5V's?

Com o advento da internet, o volume de dados aumentou bruscamente, de forma que os processamentos normais não conseguiam mais processar, surgindo o termo Big Data.

Inicialmente, eram-se considerados três aspectos, ou 3 V's para o Big Data:

- Volume: refere-se à quantidade de dados coletados, produzidos, processados pela organização.
- Velocidade: refere-se ao tempo em que os dados são gerados, bem como o tempo em que eles são processados.
- Variedade: refere-se ao tipo de dados, estruturados (como arquivos csv), semiestruturados (como arquivos json) e não estruturados (arquivos de imagens, áudios...).

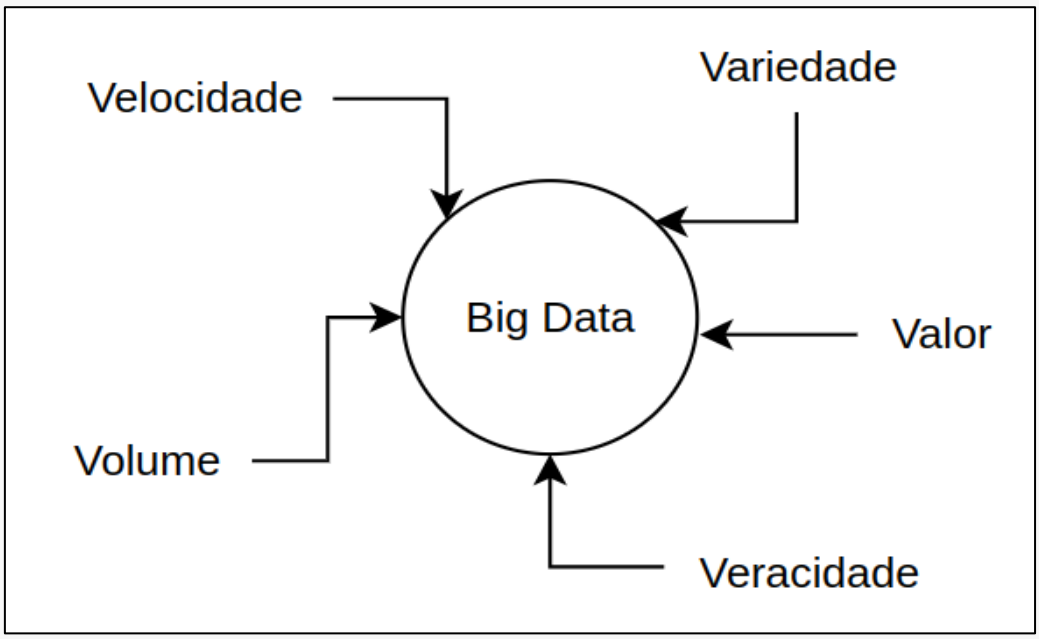
Com o tempo, percebeu-se que não importava o volume astronômico dos dados nem o quão rápido eles são gerados e processados,

independentemente de seu formato, se esses dados não agregassem valor as empresas. Com isso começaram a considerar o valor dos dados, ou seja, além dos itens citados anteriormente, os dados precisam ajudar a empresa a performar, a análise através deles deve ajudar a empresa a enxergar oportunidades ou sanar problemas que, sem eles, seria difícil e em alguns casos até impossível. Para que os dados possam agregar valor à empresa, esses dados têm que ser verdadeiros, surgindo então outro aspecto: o da Veracidade. Dessa forma, acrescentamos:

- Valor: os dados devem gerar valor para a empresa.
- Veracidade: os dados devem ser verdadeiros, reais.

Concluindo o raciocínio, big data consiste em um processo com volume gigantesco de dados, sendo esse gerados com variação e velocidades também gigantescas, onde busca extrair valor para as empresas através desses dados e assegurando que esses são verídicos. Algumas pessoas até citam outros V's, mas vamos nos ater à 5.

Figura 6 – 5V's Big Data.



> Capítulo 2

Capítulo 2. File Types e Data Types

2.1. Formatos de arquivos mais usados no pipeline de Engenharia de Dados

Comumente em diversos pipelines, arquivos utilizados como fonte de dados. Por exemplo, transações processadas por uma empresa de cartão de crédito são disponibilizadas em um formato parquet. Abaixo vamos ver alguns formatos mais utilizados.

TXT – Abreviação para Texto, consiste em arquivos legíveis para seres humanos e computadores, com dados em “palavras”, tendo esses que serem extraídos através da posição ou por alguma regra de “regex”.

CSV (Comma Separated Value) – Em português-brasileiro, significa Valores Separados por Vírgula. Consiste em um formato estruturado, onde os dados são armazenados em linhas e colunas, com geralmente a primeira linha referindo-se ao título das colunas.

JSON (JavaScript Object Notation) – Consiste em um arquivo estruturado em níveis, mas esses níveis podem variar, diferente de um arquivo estruturado que independente da coluna conter dados, a estrutura será a mesma. Teve sua origem na linguagem Java Script mas não é uma linguagem de programação.

ORC (Optimized Row Columnar) – formato otimizado de arquivo estruturado em colunas e linhas, muito mais performático que CSV ou JSON, desenvolvido pela Apache.

Avro – Pode ser considerado uma otimização do formato JSON, é o arquivo gerado pelo Apache Avro, um sistema de serialização de dados. Segundo Pinheiro Marcel, no artigo “AVRO”, o Avro ajuda a definir um formato binário para seus dados, bem como mapeá-lo para a linguagem de programação de sua escolha.

SequenceFile – Segundo Othela, no seu artigo “SequenceFile”, é um arquivo plano composto por pares de chaves binárias/valores. Ele é amplamente utilizado no Hadoop para armazenar dados, pois é comprimido e fornece acesso rápido aos registros com base em suas chaves.

Parquet – É um formato de dados estruturados em forma de colunas e linhas, porém ele é otimizado, em alguns casos, em relação a um arquivo *.csv*, pode se obter uma redução de tamanho de até 60%. Está disponível em todo o ecossistema Hadoop, desenvolvido pela fundação Apache.

Delta – Baseado no parquet, arquivo estruturado orientado por colunas, é um formato desenvolvido pela Databricks para utilização no Delta Lake. De certa forma podemos pensar num parquet com metadados, armazenando logs bem como checkpoints dos arquivos.

2.2. Data Types

Inicialmente, um dos grandes problemas de dados era o armazenamento e que muitas vezes significava o maior custo da área. Com o passar do tempo vimos novas tecnologias sendo lançadas e novos formatos de dados sendo desenvolvidos, chegando nos tempos atuais, onde o armazenamento já não é o principal ofensor ao custo total das soluções de dados. Contudo, diariamente são necessárias tratativas nos dados, mudando seus tipos, e para isso é importante saber quais são e suas diferenças, que vão além do formato, entrando ao nível de quantidade de bytes gastos para armazenar.

Tratar datatypes é uma atividade recorrente do profissional de engenharia de dados, pois muitas vezes os dados são armazenados em formato *.csv*, onde todos os tipos são no formato ‘string’, formato que pode armazenar caracteres diversos. E para que o dado tenha seu devido valor, é necessário que também esteja no seu tipo correto. Por exemplo, se em uma planilha incluirmos uma coluna de data/hora e salvarmos essa planilha em formato *.csv*, essa coluna não será mais de data/hora e na hora de

montarmos o pipeline, essa coluna deverá ser tratada em um formato timestamp, por exemplo.

Tabela 2 – Data Types.

Data Type	Value type
ByteType	int or long Note: Numbers will be converted to 1-byte signed integer numbers at runtime. Please make sure that numbers are within the range of -128 to 127.
ShortType	int or long Note: Numbers will be converted to 2-byte signed integer numbers at runtime. Please make sure that numbers are within the range of -32768 to 32767.
IntegerType	int or long
LongType	long Note: Numbers will be converted to 8-byte signed integer numbers at runtime. Please make sure that numbers are within the range of -9223372036854775808 to 9223372036854775807. Otherwise, please convert data to decimal.Decimal and use DecimalType.
FloatType	float Note: Numbers will be converted to 4-byte single-precision floating point numbers at runtime.
DoubleType	float
DecimalType	decimal.Decimal
StringType	string
BinaryType	bytearray

BooleanType	bool
TimestampTyp	datetime.datetime
DateType	datetime.date
ArrayType	list, tuple or array
MapType	dict
StructType	list or tuple
StructField	The value type in Python of the data type of this field (For example, Int for a StructField with the data type IntegerType)

Fonte: https://spark.apache.org/docs/3.0.0-preview2/sql-ref-datatypes.html#:~:text=Spark%20SQL%20and%20DataFrames%20support%20the%20following%20data,session%20local%20time-zone.%20...%206%20Complex%20types%20

> Capítulo 3

Capítulo 3. Arquiteturas

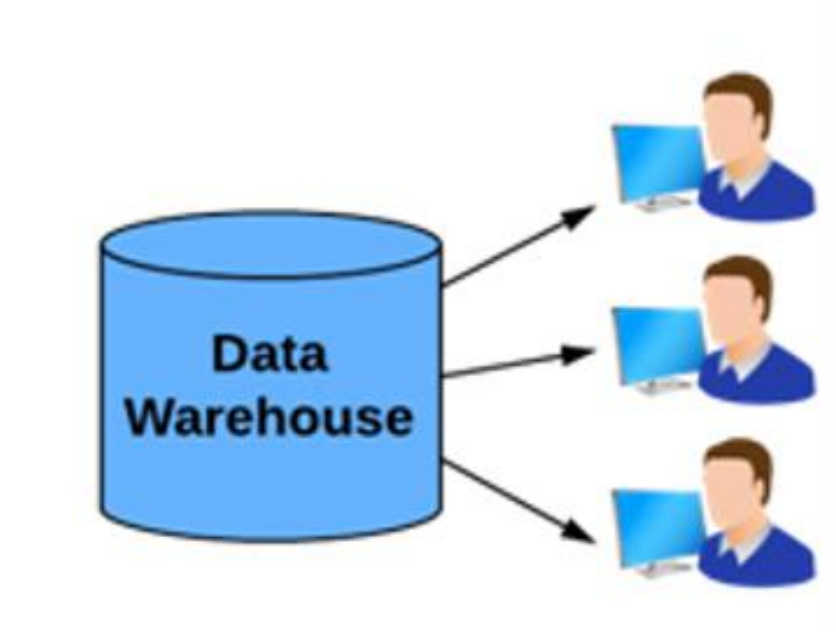
Sistemas OLAP e OLTP tem suas arquiteturas já conhecidas. Para problemas novos temos que pensar em soluções novas, e o Big Data trouxe vários problemas a serem resolvidos, alguns já o foram, outros ainda estão em discussão, contudo em relação a arquitetura de Big Data, consideraremos duas, Lambda e Kappa, mas antes vamos fazer uma revisão sobre Arquiteturas OLTP e OLAP.

3.1. OLTP

Fonte: https://arquivo.canaltech.com.br/business-intelligence/o-que-significa-oltp-e-olap-na-pratica/.

Online Transactional Processing, no português-brasileiro Processamento de Transações Online, é utilizado para as transações relacionadas ao negócio propriamente dito, transações bancárias, entrada e saída de produtos do estoque.

3.2. OLAP



Fonte: https://arquivo.canaltech.com.br/business-intelligence/o-que-significa-oltp-e-olap-na-pratica/.

Online Analytical Processing – Processamento Analítico Online. Está relacionado à análise de dados, permitindo consultas em um grande volume de dados, por exemplo, uma consulta em linguagem SQL em um Data Warehouse.

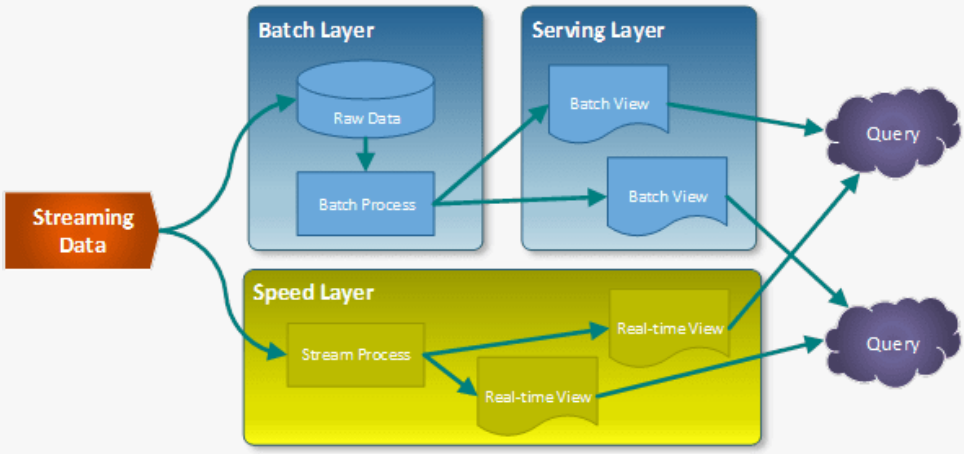
OLTP x OLAP

	OLTP	OLAP
Foco	Atua no nível operacional com foco na execução operacional.	Atua no nível estratégico com foco na tomada de decisão.
Performance	Opera com alta velocidade na manipulação de dados operacionais e não cria análises gerenciais.	Otimiza a leitura e a criação de análises e relatórios gerenciais.
Dados	Transações de OLTP são a fonte original dos dados.	Banco de dados OLTP são a fonte de dados para OLAP.
Estrutura dos dados	Modelagem relacional normalizada com alto nível de detalhes e otimizada para o uso transacional.	Modelagem dimensional com alto nível de sumarização.
Transação	Transações curtas e rápidas.	Transações longas e mais demoradas.
Abrangência	Utilizada por técnicos e analistas de diversas áreas da instituição.	Utilizada por gestores e membros do alto escalão da empresa para tomada de decisões.
Frequência de atualização	Feita no momento de transação dos dados, com alta frequência de atualização.	Feita no processo de carga dos dados e pode ser feita de acordo com os critérios da instituição (diária, semanal, mensal etc.).

Integridade	Mantém a restrição de integridade dos dados.	A integridade dos dados não é afetada, pois eles não são modificados com frequência.
Permissões nos dados	Leitura, inserção, modificação e exclusão dos dados.	Inserção e leitura para usuários liberados e apenas leitura para usuário final.

Fonte: https://blog.xpeducacao.com.br/oltp-e-olap/.

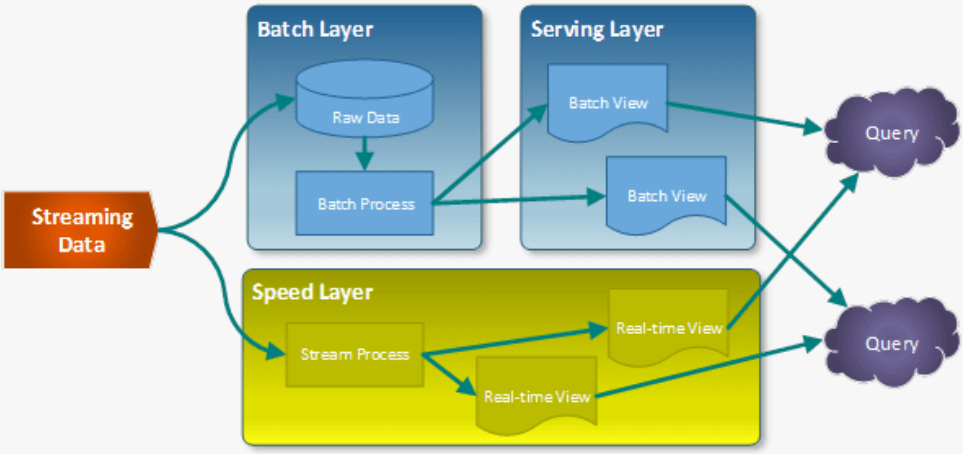
3.3. Lambda



Basicamente, ela consiste numa arquitetura com uma camada quente e uma camada fria, em outras palavras uma camada de processamento em batch e uma camada de processamento em streaming. Segundo Tejada Zoiner, no seu artigo “Arquiteturas de Big Data” (2023):

> A camada de lote alimenta uma camada de serviço que indexa a exibição de lote para uma consulta eficiente. A camada de velocidade atualiza a camada de serviço com atualizações incrementais de acordo com os dados mais recentes. Os dados que fluem para o caminho quente são restritos por requisitos de latência impostos pela camada de velocidade, de modo que ela possa ser processada o mais rapidamente possível.

Figura 7 – Arquitetura Lambda.



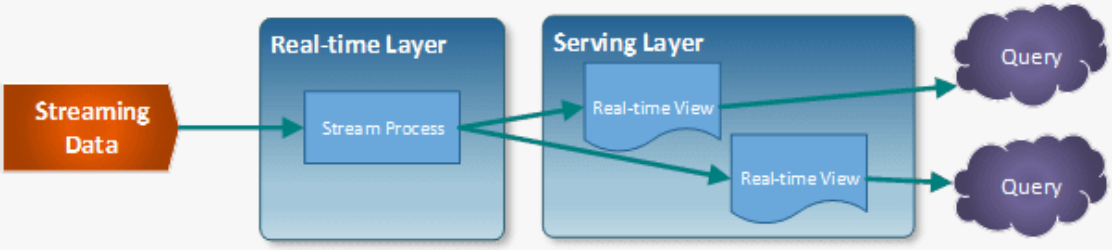
Fonte: https://www.anselme.com.br/wp-content/uploads/2023/07/image-9.png.

Kappa

É uma arquitetura mais nova que Lambda e foi desenvolvida como uma alternativa a mesma, buscando evitar a complexidade.

Nela todos os dados passam pelo mesmo fluxo, conforme Kreps Jay, citado por Junior José, sua proposta é eliminar a camada de processamento em batch, deixando apenas a camada de streaming.

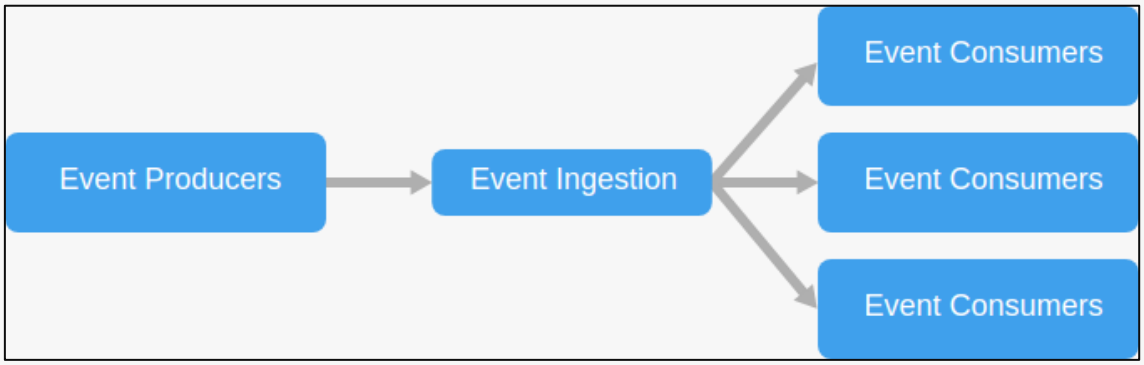
Figura 9 – Arquitetura Kappa.



Fonte: https://www.anselme.com.br/wp-content/uploads/2023/07/image-10.png.

3.4. EDA – Event-Driven Architecture

Figura 9 – Event Driven Architecture.



Fonte: https://learn.microsoft.com/pt-br/azure/architecture/guide/architecture-styles/event-driven.

Diariamente, jobs e mais jobs são executadas nas empresas contendo pipelines em streaming ou batch para processamento de dados, os jobs em streaming ficam ligados continuamente, os jobs em batch normalmente são executados em horários específicos.

Há porém um outro tipo de arquitetura onde o job é acionado por determinado acontecimento, no caso, a arquitetura orientada a eventos. Por exemplo, a chegada de um arquivo em um storage pode acionar e executar o pipeline. Essa arquitetura é separada em Produtores, que geram o fluxo, e Consumidores, que ficam ouvindo o fluxo, esperando o evento para seguir o processamento.

Modelos

Pub/Subestrutura de publicação e assinatura – Quando um evento é publicado ele é enviado para cada consumidor.

Transmissão de Eventos – Os eventos são gravados e os consumidores não precisam se inscrever em uma transmissão de evento. Na verdade, eles podem ler a partir de qualquer parte da transmissão e ingressar nela a qualquer momento. Conforme a RedHat, no artigo “What is event-driven architecture?” (2019), há alguns tipos de transmissão de eventos:

- Processamento do fluxo do evento: usa uma plataforma de transmissão, como o Apache Kafka, para ingerir e processar eventos ou transformar seu fluxo. Esse método pode ser usado para detectar padrões significativos em fluxos de eventos.

- Processamento de evento simples: é quando um evento aciona imediatamente uma ação no consumidor.

- Processamento de evento complexo: requer que um consumidor processe uma série de eventos para detectar padrões.

Frameworks

Podemos destacar como principais frameworks: o Apache Kafka, oferecido pela própria Apache é uma plataforma de streaming de eventos distribuídos de código aberto, usada por milhares de empresas para pipelines de dados de alto desempenho, análise de streaming, integração de dados e aplicativos de missão crítica; e o Google Pub/Sub, que segundo sua documentação consiste em um serviço de mensagens assíncrono e escalonável, que separa os serviços que produzem mensagens dos serviços que processam essas mensagens.

> Capítulo 4

Capítulo 4. Técnicas de Coleta de Dados

A primeira etapa de um pipeline de dados, muitas vezes, consiste na extração de dados, e dependendo do nível de abstração, se desconsideramos autenticação bem como o acionamento de máquinas de um cluster, temos alguns tipos de coletas de dados.

4.1. Crawler

Crawler ou Web Crawler, consistem em um algoritmo que varre os sites em busca de padrões e ou informações.

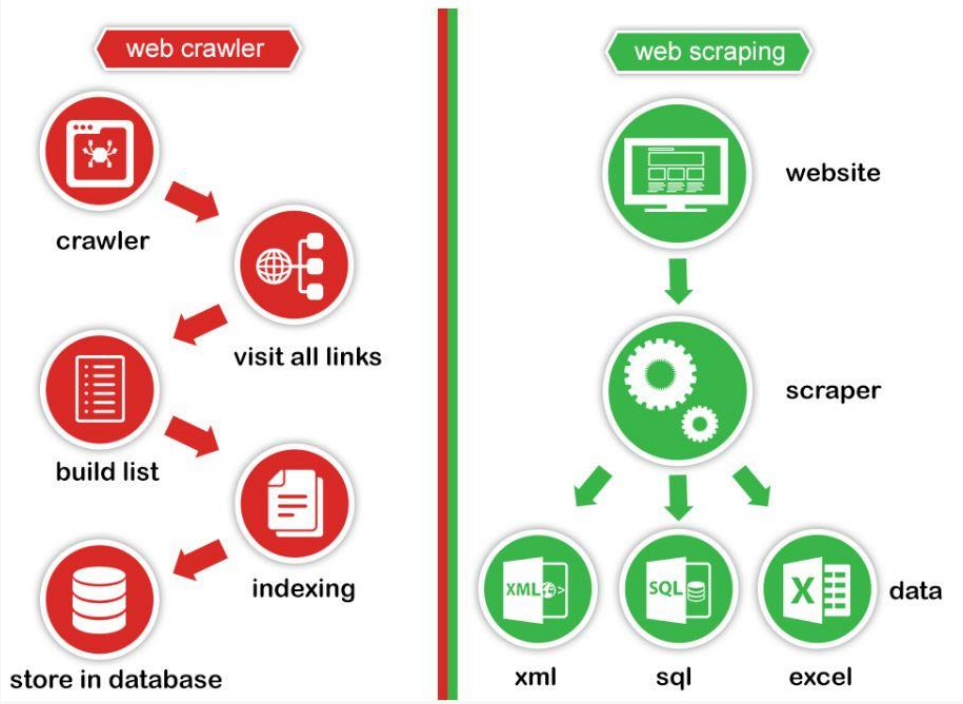
Segundo Moraes Daniel, no artigo “Web crawler: Saiba o que é e qual a sua relação com o Marketing Digital” (2018), Web crawler, ou bot, é um algoritmo usado para analisar o código de um website em busca de informações, e depois usá-las para gerar insights ou classificar os dados encontrados.

4.2. Scrapping

Screapping – ou Web Screapping. Diferente do Crawler, que é amplo e percorre todo o site, o screapping é utilizado para encontrar informações em partes específicas.

> Também chamado de raspagem web, o scraping permite coletar informações na internet de maneira automatizada a partir de bases de dados públicas disponibilizadas em sites, redes sociais e outros serviços online. O scraping é acionado quando um pesquisador, cientista, jornalista ou outro profissional precisa levantar uma grande quantidade de dados para alimentar um estudo, uma pesquisa ou uma reportagem, automatizando a coleta em uma base pública do governo federal ou de qualquer outra fonte. (Gonçalvez, 2021)

Figura 10 – Crawler x Screapping



Fonte: https://jungjihyuk.github.io/2019/07/15/data-collection/.

4.3. API – Application Programming Interface

Consiste em um mecanismo que permite de forma automatizada a comunicação entre dois componentes de software.

Por exemplo, API de Rastreio dos Correios permite que você rastreie sua encomenda através de um protocolo disponível na documentação da própria API.

Conforme AWS, no artigo “o que é uma API?”:

Existem tipos de API:

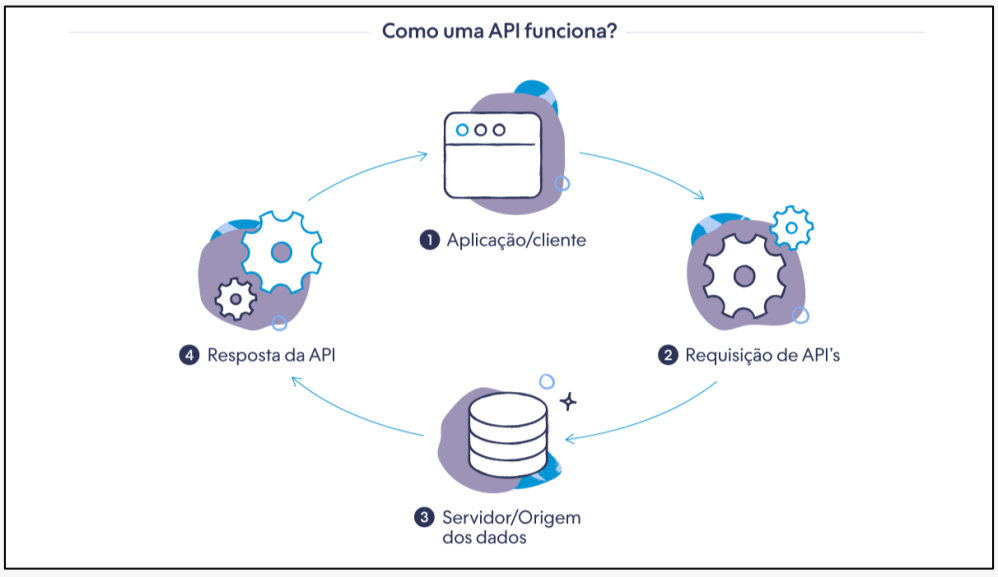
- APIs SOAP – Essas APIs usam o Simple Object Access Protocol (Protocolo de Acesso a Objetos Simples). Cliente e servidor trocam mensagens usando XML. Essa é uma API menos flexível que era mais popular no passado.

- APIs RPC – Essas APIs são conhecidas como Remote Procedure Calls (Chamadas de Procedimento Remoto). O cliente conclui uma função (ou um procedimento) no servidor e o servidor envia a saída de volta ao cliente.

- APIs WebSocket – A API de WebSocket é outro desenvolvimento de API da Web moderno que usa objetos JSON para transmitir dados. Uma API WebSocket oferece suporte à comunicação bidirecional entre aplicativos cliente e o servidor. O servidor pode enviar mensagens de retorno de chamada a clientes conectados, tornando-o mais eficiente que a API REST.

- APIs REST – Essas são as APIs mais populares e flexíveis encontradas na Web atualmente. O cliente envia solicitações ao servidor como dados. O servidor usa essa entrada do cliente para iniciar funções internas e retorna os dados de saída ao cliente.

Figura 11 – API.



Fonte: https://d1ih8jugeo2m5m.cloudfront.net/2021/05/API_Infografico4.png.

> Capítulo 5

Capítulo 5. Arquitetura de Microsserviços

5.1. Conceitos e Aplicações

Análogo ao processamento Micro Batch, que são pequenos lotes de processamento que fazem parte de um dataset maior, a arquitetura de microsserviços diz respeito a uma aplicação dividida em pequenas partes, tendo cada parte seu próprio código, facilitando a manutenção, correção, e possibilitando que algumas etapas ocorram em paralelo. Segundo Kanczuk Daniel, no artigo “O que são microsserviços e como funcionam?” (2020), Microsserviços são um tipo inovador de arquitetura de software, que consiste em construir aplicações desmembrando-as em serviços independentes.

Virtualização x containers

A virtualização é basicamente virtualizar, recursos de uma máquina, criando um ambiente separado, por exemplo: Se você tem uma máquina física com SO Windows e precisa de uma máquina com SO Linux, você pode, com uma imagem iso do Linux, através de um software de virtualização, como VitrualBox, “criar” uma máquina virtual dentro da sua máquina física.

É também muito utilizada para estudos, visto que, enquanto estamos aprendendo, muitas vezes erramos, esses erros em um sistema operacional podem ocasionar grandes problemas, estando em uma máquina virtual, basta você exclui-a e gerar uma nova, sem nenhum impacto na sua máquina local.

Segundo Buchanan (2023), temos os seguintes prós e contras em relação ao uso de VM’s:

Prós:

- Segurança com Isolamento Total.

- Desenvolvimento Interativo.

Contras:

- Velocidade de Interação;.
- Custo do tamanho do armazenamento.

Principais provedores: VirtualBox, VMWare, QEMU

Tipos:

- Virtualização de dados: dados distribuídos em locais distintos, são consolidados.
- Virtualização de desktop: cria-se máquinas virtuais compartilhando os mesmos recursos físicos disponíveis no servidor.
- Virtualização de Sistema Operacional: virtualização feita pelo Kernel, igual a virtualização de desktop, onde há compartilhamento de recursos físicos. Contudo, agora são em máquinas locais e não servidores.
- Virtualização de Funções de rede: separa funções de rede para distribui-las, ao invés de ter que montar outras redes fisicamente.

5.2. Containers

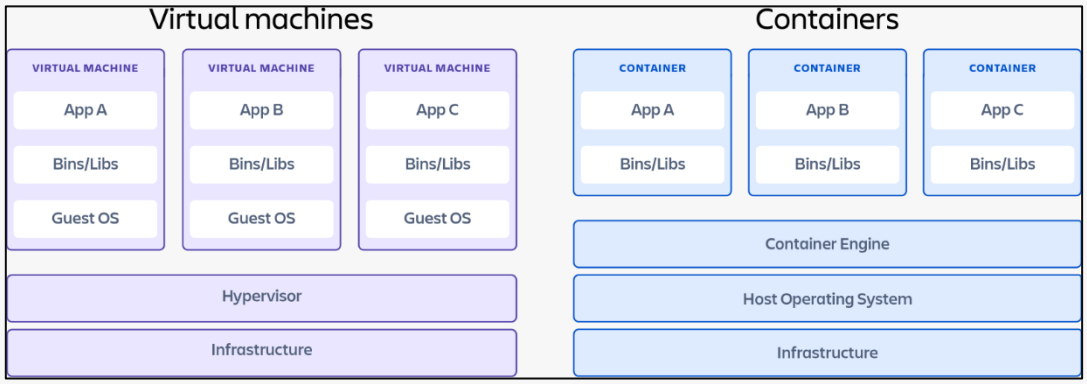
Conteinirização consiste no empacotamento de todos os itens necessários de uma aplicação, como código fonte, bibliotecas e frameworks, para que esses fiquem isolados, garantido a independência e a correta execução.

Conforme RedHat nos fala no artigo: “O que é Conteinerização” (2023), enquanto as máquinas virtuais funcionam bem com arquiteturas de TI monolíticas tradicionais, os containers foram criados para serem

compatíveis com tecnologias emergentes mais novas, como nuvens, CI/CD e DevOps.

> Contêineres e máquinas virtuais são tecnologias de virtualização de recursos muito semelhantes. A virtualização é o processo no qual um recurso singular do sistema, como RAM, CPU, Disco ou Rede, pode ser "virtualizado" e representado como vários recursos. O principal diferencial entre contêineres e máquinas virtuais é que as VMs virtualizam uma máquina inteira até as camadas de hardware, enquanto os contêineres virtualizam apenas camadas de software acima do nível do sistema operacional. (Bunchanan, 2023)

Figura 12 – Containers x Virtual Machines



Fonte: https://www.atlassian.com/br/microservices/cloud-computing/containers-vs-vms.

5.3. Docker e Kubernetes: conceitos básicos

Docker – segundo o site da própria Docker Inc., é uma plataforma aberta para desenvolvimento, envio e execução de aplicativos. O Docker permite que você separe seus aplicativos de sua infraestrutura para que você possa fornecer software rapidamente. Com o Docker, você pode gerenciar sua infraestrutura da mesma forma que gerencia seus aplicativos.

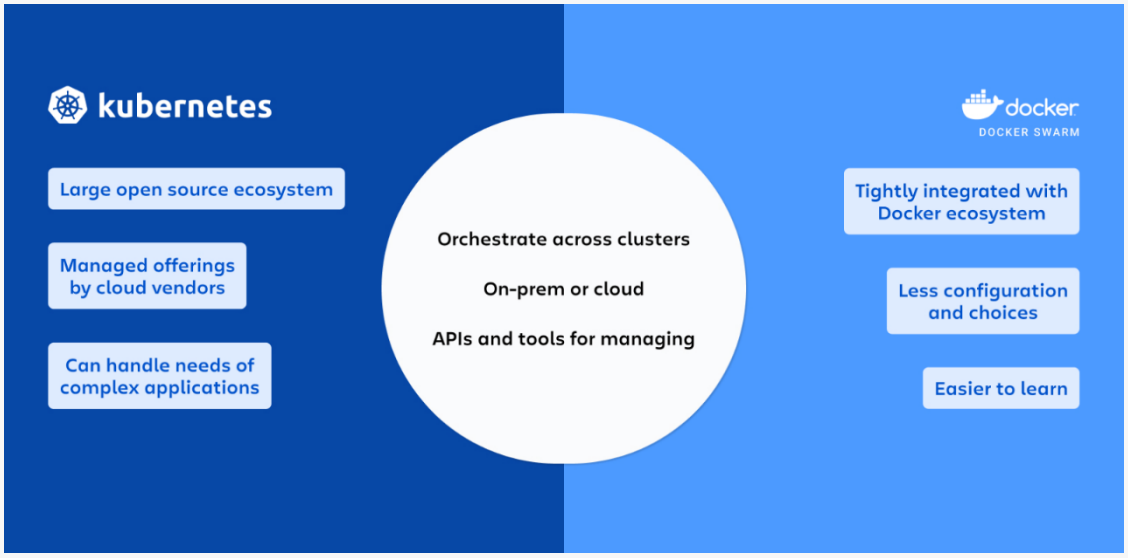
Docker Compose – Utilizado para definir e executar containers, configurando através de um arquivo yaml. Após a configuração, com o comando “docker compose up” se inicia todos os serviços da configuração.

Kubernets – conforme o site do Kubernets, é uma plataforma portátil, extensível e de código aberto para gerenciar cargas de trabalho e serviços em contêineres, que facilita a configuração declarativa e a automação. Tem um ecossistema grande e em rápido crescimento. Os serviços, suporte e ferramentas do Kubernetes estão amplamente disponíveis.

O Kubernetes gerencia automaticamente coisas como descoberta de serviços, balanceamento de carga, alocação de recursos, isolamento e escalabilidade vertical ou horizontal de pods (CAMPBELL).

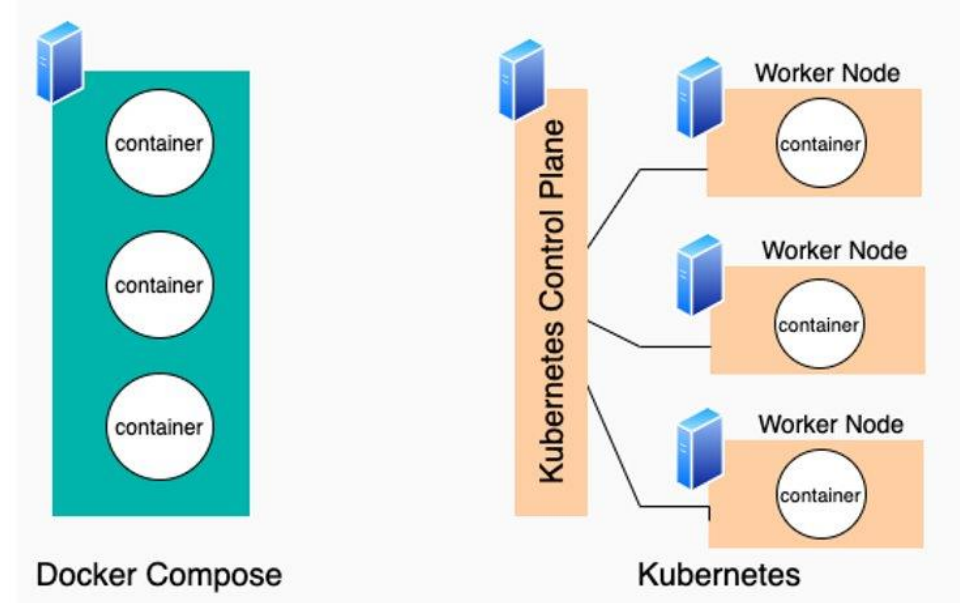
Dessa forma, vemos que Docker e K8s não são concorrentes e sim complementares, onde o Docker é utilizado para isolar sua aplicação em containers e o K8s utilizado para executar e gerenciar os containers.

Figura 13 – Kubernets x Docker.



Fonte: https://www.atlassian.com/br/microservices/microservices-architecture/kubernetes-vs-docker.

Docker Compose x Kubernets



Fonte: https://www.theserverside.com/blog/Coffee-Talk-Java-News-Stories-and-Opinions/What-is-Kubernetes-vs-Docker-Compose-How-these-DevOps-tools-compare.

> Capítulo 6

Capítulo 6. Data Governance

6.1. Governança de Dados

Governança de dados é a subárea dentro da área de dados que tem a função de gerenciar os dados.

> Gestão de Dados é a função na organização que cuida do planejamento, controle e entrega de ativos de dados e de informação. Esta função inclui: as disciplinas do desenvolvimento, execução e supervisão de planos, políticas, programas, projetos, processos, práticas, e procedimentos que controlam, protegem, distribuem e aperfeiçoam o valor dos ativos de dados e informações". E a Governança de Dados consiste no exercício de autoridade e controle (planejamento, monitoramento e execução) sobre o gerenciamento de ativos de dados. (DMBOK, 2012)

> É um procedimento de tomada de decisões e responsabilidades para com os processos relacionados aos dados, baseando-se em políticas, normas e restrições. O foco de atuação [do programa] pode variar de organização para organização, mas para ser estruturada e eficiente é preciso que as organizações definam suas necessidades de gestão de dados, bem como os objetivos a serem atingidos, e a partir deste ponto, delimitem o escopo de atuação. (FERNANDES; ABREU, 2012)

6.2. DAMA

Roda Dama

Elenca as áreas de conhecimento da Gestão dos Dados, colocando a Governança de Dados no centro, pois ela é responsável pelo perfeito equilíbrio.

Fonte: https://jkolb.com.br/a-estrutura-do-dama-dmbok/.

6.3. Dama DMBOK

Hexágono de Fatores Ambientais:

- Mostra a relação entre pessoas, tecnologia e processos.
- Metas e Princípios estão no centro pois são o foco.

Ferramentas e técnicas são como as pessoas vão executar atividades entregáveis e de acordo com suas responsabilidade e papéis, seguindo a organização da empresa para o sucesso no gerenciamento dos dados.

Fonte: https://jkolb.com.br/a-estrutura-do-dama-dmbok/.

> Capítulo 7

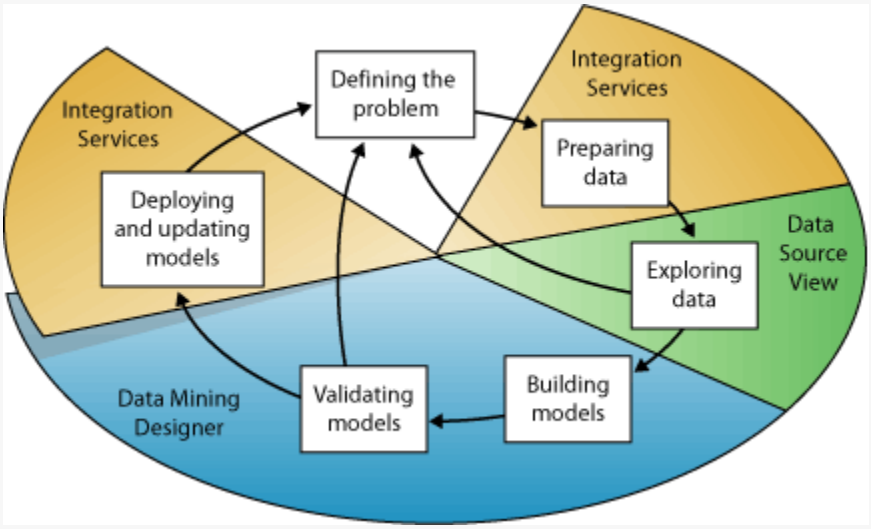
Capítulo 7. Mineração de dados

Mineração de dados, de certo ponto de vista, é um termo que não reflete a realidade, pois na verdade, consiste em minerar informações. Para um melhor entendimento podemos fazer uma analogia com a mineração de pedras preciosas.

- Os dados são a terra, pedras sem valor de mercado e demais materiais que existam no solo;
- As informações são as pedras preciosas que com trabalho e esforço você consegue as encontrar.

Já um exemplo na área de Engenharia de Dados. No universo dos dados de compras realizadas em um site de marketplace, você quer encontrar um padrão que dos clientes para compra de determinados produtos. Existem diversas técnicas e algoritmos aos quais nos limitaremos a alguns.

Figura 14 – Mineração de Dados



Fonte: https://learn.microsoft.com/pt-br/analysis-services/data-mining/data-mining-concepts?view=asallproducts-allversions.

De uma maneira macro, vamos avaliar a imagem que mostra o passo a passo do desenvolvimento de uma solução de mineração de dados.

Inicialmente se define o problema a ser respondido com a solução.

Após a definição do problema, começa a preparação e a exploração dos dados, pode ser que estejam em outros ambientes, sendo necessário integrá-los. Acontece de os dados estarem em granularidades diferente, entre outras particularidades que exigirão tratamento e/ou limpeza. Os dados prontos para consumo, e previamente explorados, verifica se serão úteis para a resolução do problema;

De posse dos dados, inicia-se a construção do modelo;

Com o modelo desenvolvido, ocorre a etapa de validação, onde são definidos parâmetros de aceitação do resultado.

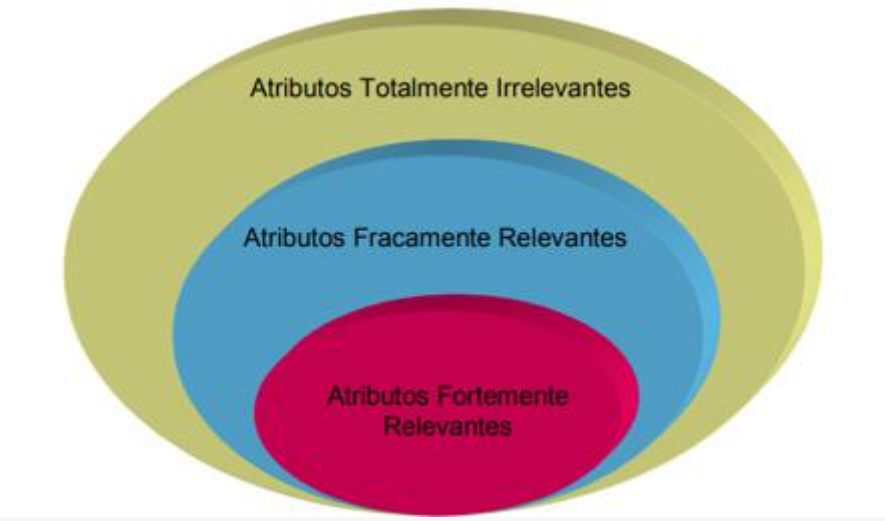
Por fim, com as etapas anteriores concluídas com sucesso, o modelo estará pronto para entrar em produção.

Uma outra etapa, que não consta na imagem, mas de muita importância, é o retreino do modelo. Atividade que demanda expertise do código, bem como da modelagem, pois consiste em ajustar parâmetros para otimizar o resultado do modelo.

7.1. Pré-processamento de dados: limpeza, integração e transformação

Etapa que consiste em realizar tratamentos, filtros e enriquecimentos nos dados buscando melhorar a qualidade dos dados nos quesitos completude e confiabilidade. De forma a não impactar a performance dos modelos de aprendizagem de máquina.

7.2. Seleção de atributos



Assim como a qualidade dos dados é de extrema importância, a seleção de atributos também é, e de certa forma, podemos dizer que é a etapa mais delicada, pois exige um conhecimento muito grande de negócios, e não só dos dados ou dos modelos.

É nessa etapa que são escolhidos os atributos ou características que serão utilizadas para treinar o modelo em busca da resposta.

Alguns modelos possuem a seleção de atributos de forma intrínseca, ou seja, embutida no seu próprio código

7.3. Modelos

Podemos agrupar modelos de Mineração de dados de acordo com o resultado que eles buscam alcançar, sendo as seguintes:

Modelagem descritiva – baseia-se em dados históricos, buscando encontrar a causa de sucesso ou fracasso de uma ação de marketing, por exemplo. Como exemplo de técnicas desse modelo, temos:

Clustering	Agrupa registros semelhantes.
Detecção de anomalias	Identifica valores discrepantes multidimensionais.
Regras de associação	Detecta relações entre registros.
Análise do componente principal	Detecta relações entre variáveis.
Grupos de afinidade	Agrupa pessoas com interesses ou objetivos semelhantes (ex., pessoas que compram X podem comprar Y e, possivelmente, Z).

Fonte: https://www.sas.com/pt_br/insights/analytics/mineracao-de-dados.html.

Modelagem preditiva – busca estimar o futuro, por exemplo, a probabilidade de um cliente não pagar o empréstimo com base em dados de pagamentos.

Regressão	Uma medida da força da relação entre uma variável dependente e uma série de variáveis independentes.
Redes neurais	Programas de computadores que detectam padrões, fazem previsões e aprendem disso.
Árvores de decisão	Diagramas na forma de árvores em que cada galho representa uma ocorrência provável.
Máquinas de vetores de suporte	Modelos de aprendizagem supervisionada com seus algoritmos de aprendizagem associados.

Fonte: https://www.sas.com/pt_br/insights/analytics/mineracao-de-dados.html

Modelagem prescritiva – avalia, analisa e busca informações para prescrever o que ocorrerá, por exemplo, prescrever a quantidade de vendas em uma black friday.

Análises preditivas e suas regras	Desenvolve regras do tipo se/então a partir de padrões e prevê resultados.
Otimização de marketing	Simula, em tempo real, o mix de mídia mais vantajoso para alcançar o maior ROI possível.

Fonte: https://www.sas.com/pt_br/insights/analytics/mineracao-de-dados.html.

> Capítulo 8

Capítulo 8. Data Governance

8.1 Manifesto Ágil

Ao acessar o site do Agile Manifesto nos deparamos com a seguinte mensagem:

> Estamos descobrindo maneiras melhores de desenvolver softwares fazendo-o nós mesmos e ajudando outros a fazerem o mesmo. Através deste trabalho, passamos a valorizar:
>
> **Indivíduos e interações** mais que processos e ferramentas.
>
> **Software em funcionamento** mais que documentação abrangente.
>
> **Colaboração com o cliente** mais que negociação de contratos.
>
> **Responder a mudanças** mais que seguir um plano. Ou seja, mesmo havendo valor nos itens à direita, valorizamos mais os itens à esquerda.

O Manifesto Ágil consiste em um documento assinado por 17 desenvolvedores de software que chegaram num senso comum sobre o processo de desenvolvimento de software, que nas empresas, devido à complexidade, demoravam muito tempo para fazer uma entrega, gerando um custo muito alto na relação desenvolvimento/entrega.

Os desenvolvedores

- Estamos descobrindo maneiras melhores de desenvolver softwares, fazendo-o nós mesmos e ajudando outros a fazerem o mesmo. Através deste trabalho, passamos a valorizar:
- Indivíduos e interações mais que processos e ferramentas.
- Software em funcionamento mais que documentação abrangente.
- Colaboração com o cliente mais que negociação de contratos.

- Responder a mudanças mais que seguir um plano. Ou seja, mesmo havendo valor nos itens à direita, valorizamos mais os itens à esquerda.

  No manifesto, foram definidos 12 princípios, sendo eles:

  1. Nossa maior prioridade é satisfazer o cliente através da entrega contínua e adiantada de software com valor agregado.

  2. Mudanças nos requisitos são bem-vindas, mesmo tardiamente no desenvolvimento. Processos ágeis tiram vantagem das mudanças visando vantagem competitiva para o cliente.

  3. Entregar frequentemente software funcionando, de poucas semanas a poucos meses, com preferência à menor escala de tempo.

  4. Pessoas de negócio e desenvolvedores devem trabalhar diariamente em conjunto por todo o projeto.

  5. Construa projetos em torno de indivíduos motivados. Dê a eles o ambiente e o suporte necessário e confie neles para fazer o trabalho.

  6. O método mais eficiente e eficaz de transmitir informações para e entre uma equipe de desenvolvimento é através de conversa face a face.

  7. Software funcionando é a medida primária de progresso.

  8. Os processos ágeis promovem desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente.

9. Contínua atenção à excelência técnica e bom design aumenta a agilidade.

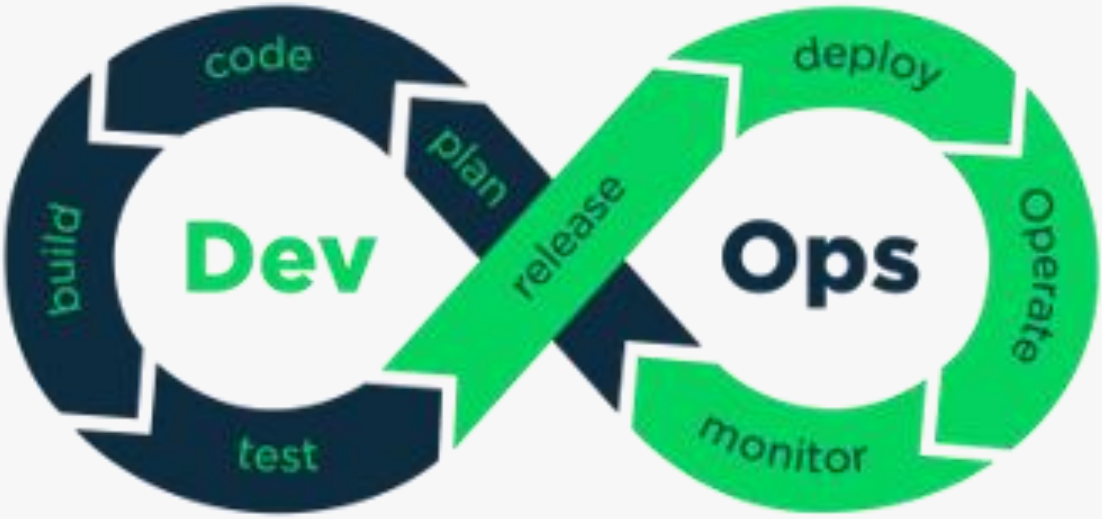
10. Simplicidade--a arte de maximizar a quantidade de trabalho não realizado--é essencial.

11. As melhores arquiteturas, requisitos e designs emergem de equipes auto-organizáveis.

12. Em intervalos regulares, a equipe reflete sobre como se tornar mais eficaz e então refina e ajusta seu comportamento de acordo.

Retirado de: Princípio por trás do Manifesto Ágil.

8.2. Devops



Fonte: https://blog.4linux.com.br/beneficios-do-devops/

Com o impacto do Big Data, assim como foram necessários desenvolver novos meios de processar e armazenar esses dados, considerando os 5V's, também se fez necessário pensar em novas formas de desenvolver, não no sentido ferramentas ou estrutura de código, mas sim, ligado a cultura.

O Manifesto Ágil trouxe várias reflexões para as empresas, no qual resultou na criação de uma nova cultura de desenvolvimento, o DevOps, que é a junção de Desenvolvimento + Operações.

Segundo a Red Hat (2018):

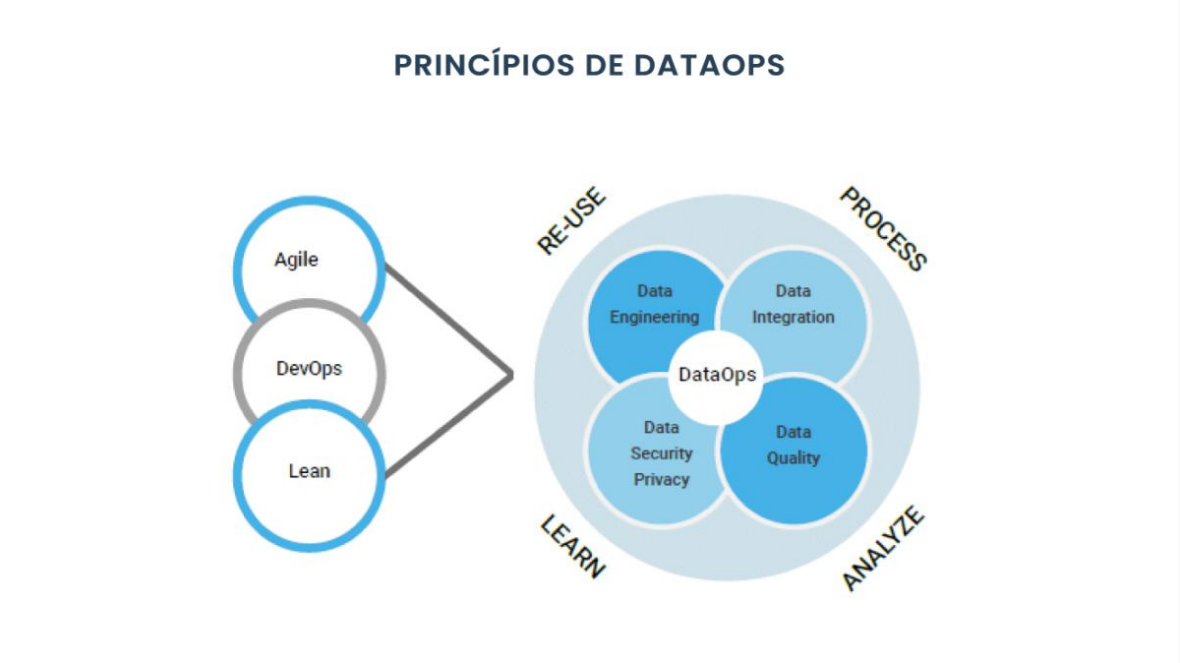
> DevOps é uma abordagem de cultura, automação e design de plataforma que tem como objetivo agregar mais valor aos negócios e aumentar a capacidade de resposta às mudanças por meio de entregas de serviços rápidas e de alta qualidade. Isso tudo é possível por meio da disponibilização de serviços de TI iterativa e rápida. Adotar o DevOps significa conectar aplicações legadas a uma infraestrutura e aplicações modernas e nativas em nuvem.

DevOps está alinhado a utilização também da aplicação de algumas metodologias de desenvolvimento, como Containerização, Arquitetura de Microsserviços, Arquitetura Orientada a Eventos.

8.3. DataOps

Dataops consiste na junção das Culturas Agile e DevOps, com práticas relacionadas ao Lean Manufacturing. Buscando

Através do trabalho com dados em organizações, com ferramentas, e em indústrias, nós pudemos desvendar a melhor maneira de desenvolver e entregar análises, que nós chamamos de DataOps.



- Indivíduos, interações sobre processos e ferramentas.
- Trabalho de análise sobre uma documentação abrangente.
- Colaboração do cliente sobre negociação de contratos.
- Experimentação, iteração, e resposta sobre um projeto detalhado e extenso.
- Propriedade de todas as equipes nas operações sobre silos de responsabilidades.

Conforme a Triggo.ai, podemos resumir os princípios do DataOps como:

1. Satisfaça continuamente o seu cliente:

Nossa maior prioridade, é satisfazer o cliente por meio da entrega antecipada e contínua de insights analíticos e valiosos de alguns minutos a semanas.

2. Foco em criar análises relevantes:

Acreditamos que a principal medida do desempenho da análise de dados é o grau em que a análises relevantes são fornecidas, incorporando dados precisos, sobre estruturas e sistemas robustos.

3. Abrace a mudança:

Acolhemos as necessidades dos clientes em evolução e, de fato, as adotamos para gerar vantagem competitiva. Acreditamos que o método de comunicação mais eficiente, eficaz e ágil com os clientes é a conversa face a face.

4. É um esporte coletivo:

As equipes analíticas sempre terão uma variedade de funções, habilidades, ferramentas favoritas e títulos. Uma diversidade de origens e opiniões aumenta a inovação e a produtividade.

5. Interações diárias:

Clientes, equipes analíticas e operações devem trabalhar juntos diariamente durante todo o projeto.

6. Auto-organização:

Acreditamos que os melhores insights analíticos, algoritmos, arquiteturas, requisitos e designs surgem de equipes auto-organizadas.

7. Reduzir o heroísmo:

À medida que o ritmo e a amplitude da necessidade de insights analíticos aumentam, acreditamos que as equipes analíticas devem se esforçar para reduzir o heroísmo e criar equipes e processos de análise de dados sustentáveis e escaláveis.

8. Refletir:

As equipes analíticas devem ajustar seu desempenho operacional refletindo, em intervalos regulares, sobre o feedback fornecido por seus clientes, por eles mesmos e pelas estatísticas operacionais.

9. Analytics é código:

As equipes analíticas usam uma variedade de ferramentas individuais para acessar, integrar, modelar e visualizar dados. Fundamentalmente, cada uma dessas ferramentas gera código e configuração que descrevem as ações realizadas nos dados para fornecer insights.

10. Orquestrar:

A orquestração do início ao fim de dados, ferramentas, código, ambientes e o trabalho das equipes analíticas é um fator-chave para o sucesso analítico.

11. Torná-lo reproduzível:

Resultados reproduzíveis são necessários e, portanto, nós versionamos tudo: dados, configurações de hardware e software de baixo nível e o código e a configuração específicos para cada solução na stack de ferramentas.

12. Ambientes sandbox disponíveis:

Acreditamos que é importante minimizar o custo de experimentação dos membros da equipe analítica, oferecendo ambientes técnicos fáceis de criar, isolados, seguros e disponíveis que reflitam seu ambiente de produção.

13. Simplicidade:

Acreditamos que a atenção contínua à excelência técnica e ao bom design aumenta a agilidade; da mesma forma, a simplicidade, a arte de maximizar a quantidade de trabalho não feito é essencial.

14. Analytics é manufatura:

Os pipelines analíticos são análogos às linhas de manufatura enxuta. Acreditamos que um conceito fundamental de DataOps é o foco no pensamento de processos visando alcançar eficiências contínuas na fabricação de insights analíticos.

15. A qualidade é primordial:

Os pipelines analíticos devem ser construídos com uma base capaz de detecção automatizada de anormalidades e problemas de segurança no código, configuração e dados, e devem fornecer feedback contínuo aos operadores para evitar erros.

16. Monitore a qualidade e o desempenho:

Nosso objetivo é ter medidas de desempenho, segurança e qualidade monitoradas continuamente para detectar variações inesperadas e gerar estatísticas operacionais.

17. Reúso:

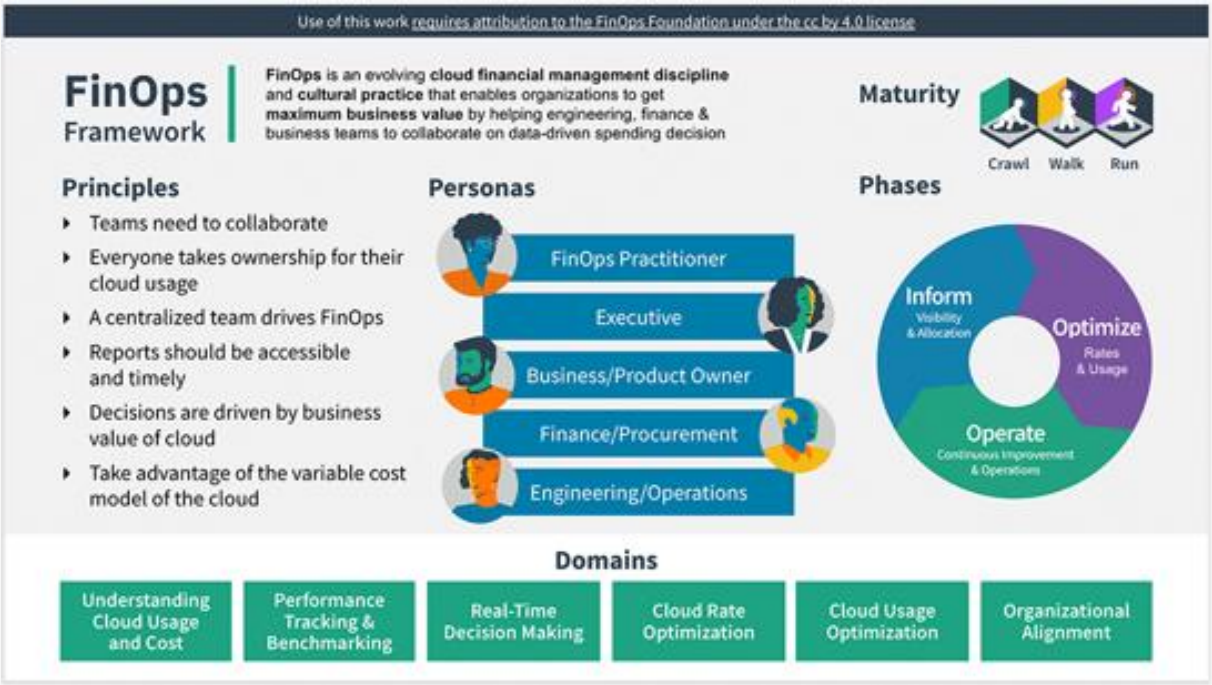
Acreditamos que um aspecto fundamental da eficiência de fabricação de insights analíticos é evitar a repetição de trabalhos anteriores do indivíduo ou da equipe.

18. Melhore os tempos de ciclo:

Devemos nos esforçar para minimizar o tempo e o esforço para transformar uma necessidade do cliente em uma ideia analítica, criá-la em desenvolvimento, lançá-la como um processo de produção repetível e, finalmente, refatorar e reutilizar esse produto.

8.4. FinOps

Uma das grandes vantagens da computação em nuvem é a escalabilidade, escalar cluster para atender uma demanda maior de dados para processamento, escalar um storage para armazenar mais dados entre outros recursos, que se fosse "on-premises" - com os recursos físicos na empresa, demandaria mais espaço para instalar os servidores entre outras desvantagens. Porém tudo tem um preço, cloud computing se não bem gerenciada, pode custar mais do que o retorno de seus produtos e aplicações.



Fonte: https://www.finops.org/introduction/what-is-finops/.

FinOps assim como DevOps e DataOps traz orientações para uma gestão efetiva dos custos relacionados a sua computação em nuvem. Tendo com princípios:

- As equipes precisam colaborar.
- Todos assumem a propriedade.
- Uma equipe centralizada impulsiona o FinOps.
- Relatórios devem ser acessíveis e oportunos.
- As decisões são orientadas pelo valor comercial da nuvem.
- Aproveite o modelo de custo variável da nuvem.

Seguindo esses princípios, descentralizando a visão dos custos, tornando as áreas parte do processo de gestão, as pessoas se sentem donas de fato do processo, e zelando por esses.

FinOps não preza por “cortar custos a qualquer custo”, isso é uma visão um pouco distorcida. Com a aplicação, busca-se de forma inteligente reduzir custos, buscando oportunidades que não impactarão nas aplicações, mas sim nos custos relacionados a esta. Por exemplo:

Alterar o tipo de máquinas no cluster, se sua aplicação consome muita memória, procurar máquinas que tenham mais memória do que processadores e assim por diante.

> Capítulo 9

Capítulo 9. Modern Data Stack

9.1. Data Mesh

Com o passar dos anos vimos as arquiteturas de armazenamento de dados mudando, algumas caindo em desuso, outras permanecem em constante utilização, dentre as arquiteturas, podemos mencionar, Data Marts, Data Warehouse, Data Lake, Lakehouse e Delta Lake.

Com a crescente utilização do self-service BI, o data mesh surgiu com a proposta de descentralização, possibilitando as diversas áreas acessarem os dados sem ter que migrarem eles para um outro ambiente.

Simplificando, Data Mesh torna os dados acessíveis, disponíveis, detectáveis, seguros e interoperáveis. O acesso mais rápido aos dados se traduz diretamente em um tempo de retorno mais rápido sem a necessidade de transporte de dados (MATOS, 2022).

9.2. Amazon Aurora zero-ETL

Astro Python SDK

O FrameWork Apache Airflow é uma ferramenta muito utilizada e conceituada no mercado de Engenharia de Dados, a cada dia, mais e mais empresas optam por ele como orquestrador dos seus pipelines, que no Airflow são chamados de DAG's - Data Acyclic Graphic.

No início de 2023, a Astronomer lançou o Astro Python SDK, uma solução em nuvem que visa diminuir o tempo de desenvolvimento das DAG's, simplificando o código e facilitando o gerenciamento, bem como outras features, e com o seguinte slogan "Let your team focus on your business". Um exemplo de facilidade é o uso do "load file". Em um processo comum do Airflow sem o Astro, para armazenar os dados de um arquivo em uma tabela, temos que:

- Importar arquivo.
- Entender a estrutura.
- Criar um Dataframe.
- Escrever os dados do Dataframe na tabela de Destino.
- Com o Astro, os passos caem pela metade:
    - Importar arquivo.
    - Escrever os dados na tabela.
    - Pois ele já entende a estrutura, infere o schema dos dados e os salva na tabela.

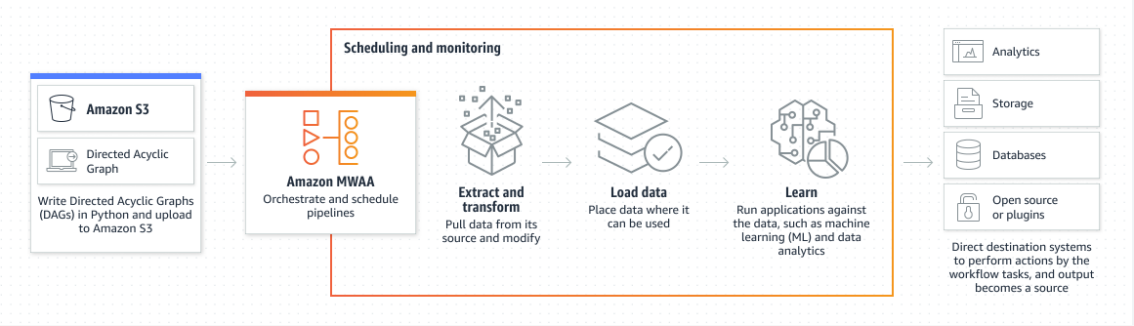
9.3. MWAA – Amazon Managed Workflows for Apache Airflow

MWAA é um serviço gerenciado do Apache Airflow, aonde o engenheiro irá se preocupar basicamente só com o código da DAG, pois toda a infraestrutura do pipeline fica sob gestão da AWS.

Algumas características que a AWS destaca são:

- Facilidade de deploy.
- Bult-in security.
- Baixo custo de operação.
- Monitoramento do Workflow na AWS ou On-Premises.
- Automatic scalling.
- Conectores integrados.

Figura 15 – Arquitetura MWAA.



Fonte: https://aws.amazon.com/managed-workflows-for-apache-airflow/.

> Capítulo 10

Capítulo 10. Boas Práticas

10.1. Clean Code

Clean Code ou código limpo em português-brasileiro, não é necessariamente uma regra ou norma técnica de desenvolvimento, mas sim, segundo Tomazel (2023), é um termo que diz respeito a um termo utilizado para descrever características do código de um software, sendo elas facilidade de leitura, entendimento, manutenção e teste.

Com o código limpo, busca-se evitar seguintes situações:

- Código com funções longas e complexas de difícil compreensão, teste e manutenção.
- Código com convenções de nomes ruins, como variáveis com nomes curtos, não descritivos ou funções com nomes que não refletem com precisão sua finalidade.
- Código com estilo inconsistente, como indentação ou espaçamento inconsistentes, o que pode dificultar a leitura e a compreensão.
- Código com código desnecessário ou redundante, o que pode dificultar a manutenção e a compreensão.
- Código que torna difícil a automação de testes, por exemplo, acoplamento (coupling) estático entre classes. (TOMAZEL, 2023)

10.2. Versionamento

Diz respeito a desenvolver um código, e a cada alteração gerar uma nova versão para ele, mas sem excluir o anterior.

> O versionamento é o processo de criar novas versões de um código toda vez que existir uma mudança significativa nele. De

> maneira geral, todo projeto de desenvolvimento é feito por etapas, sendo que as funcionalidades são incrementadas aos poucos. Por isso, é preciso criar versões que possam ser retomadas sempre que necessário. (Code, 2022)

Atualmente, existem vários frameworks para essa demanda, alguns Cloud Providers, como a AWS, já possuem seu próprio serviço de versionamento de código, no caso, o Code Commit. Os mais utilizados no mercado são Github e GitLab.

Além do controle de versão, uma outra vantagem de utilizar essa prática é a possibilidade de construir “esteiras” de CICD – Continuous Improvement Continuos Deployment, prática que permite de forma ágil subir códigos para produção, mas isso é uma conversa para outro contexto.

10.3. Documentação

Mais do que incluir comentários nas funções e classes de um código em python conforme orienta a PEP 8 por exemplo, documentar consiste em escrever de forma técnica, mas ainda sim legível para quem não conheça de linguagem de programação, o que acontece no pipeline de dados, quais as tratativas dos dados, como é integrado, como é exportado, quais tabelas consomem esses dados, quais as versões das bibliotecas utilizadas no código, desenho arquitetural do pipeline de dados.

Um framework bastante utilizado para documentação é o Confluence, nele é possível armazenar imagens, códigos diagramas, fluxogramas, vincular páginas e demais itens que facilitam o entendimento da documentação.

Referências

AMORIN, Claudio, et al. 2009. Virtualização: modelos, técnicas e exemplos de uso na construção de serviços web. Disponível em: https://www.researchgate.net/publication/282027067_Virtualizacao_modelos_tecnicas_e_exemplos_de_uso_na_construcao_de_servicos_web. Acesso em: 18 ago. 2023.

ARBIT, Modern Data Warehouse: entenda tudo sobre esse conceito, 2019. Disponível em: https://blog.arbit.com.br/modern-data-warehouse-entenda-conceito/. Acesso em: 18 ago. 2023.

AVILA, Ricardo, A Importância do Pré-processamento de Dados, 2020. Disponível em: https://theavila.github.io/2020/07/27/preprocessing/. Acesso em: 18 ago. 2023.

AVILA, Ricardo, Classificação de Flores do Tipo Iris, 2020. Disponível em: https://theavila.github.io/2020/07/04/adaboost-IRIS/. Acesso em: 18 ago. 2023.

AWS, O que é ETL zero? ,Disponível em: https://aws.amazon.com/pt/what-is/zero-etl/#:~:text=ETL%20zero%20utiliza%20tecnologias%20de,e%20de%20processamento%20de%20dados. Acesso em: 18 ago. 2023.

AWS, O que é uma malha de dados?, Disponível em: https://aws.amazon.com/pt/what-is/data-mesh/. Acesso em: 18 ago. 2023.

BARANAUSKAS, José, Seleção de Atributos FSS, Disponível em: https://dcm.ffclrp.usp.br/~augusto/teaching/ami/AM-I-FSS.pdf. Acesso em: 18 ago. 2023.

BECK, Kent, et al. Manifesto para Desenvolvimento Ágil de Software. Disponível em: https://agilemanifesto.org/iso/ptbr/manifesto.html. Acesso em: 18 ago. 2023.

BUCHANAN, Ian, Contêineres vs. máquinas virtuais. Disponível em: https://www.atlassian.com/br/microservices/cloud-computing/containers-vs-vms. Acesso em: 18 ago. 2023.

CIENCIAEDADOS, O Que é Data Mesh?. Disponível em: https://www.cienciaedados.com/o-que-e-data-mesh/. Acesso em: 18 ago. 2023.

CONFLUENT, What Is Event-Driven Architecture?. Disponível em: https://www.confluent.io/learn/event-driven-architecture/. Acesso em: 18 ago. 2023.

Data Science Academy. O que é DataOps? Um Exemplo de Caso de Uso, 2023. Disponível em: https://blog.dsacademy.com.br/o-que-e_dataops/. Acesso em: 18 ago. 2023.

DIALHOST INTERNET, VMS ou Containers: Entenda quais as diferenças e seus usos, 2020. Disponível em: https://www.dialhost.com.br/blog/vms-ou-containers/. Acesso em: 18 ago. 2023.

DIDÁTICA TECH, Como funciona o algoritmo Árvore de Decisão. Disponível em: https://didatica.tech/como-funciona-o-algoritmo-arvore-de-decisao/. Acesso em: 18 ago. 2023.

DWTOBIGDATA, Modern Data Warehouse Architecture. Disponível em: https://dwtobigdata.wordpress.com/2015/08/18/modern-data-warehouse-architecture/. Acesso em: 18 ago. 2023.

IBM, O que é arquitetura orientada por eventos?. Disponível em: https://www.ibm.com/br-pt/topics/event-driven-architecture. Acesso em: 18 ago. 2023.

IBM, O que é Mineração de Dados. Disponível em: https://www.ibm.com/br-pt/topics/data-mining. Acesso em: 18 ago. 2023.

ICHI.PRO, 4 arquiteturas de big data, fluxo de dados, arquitetura Lambda, arquitetura Kappa e arquitetura Unifield. Disponível em: https://ichi.pro/pt/4-arquiteturas-de-big-data-fluxo-de-dados-arquitetura-lambda-arquitetura-kappa-e-arquitetura-unifield-239404688940729. Acesso em: 18 ago. 2023.

JOHANSON, Lovisa, Part 1: Apache Kafka for beginners - What is Apache Kafka?, 2020. Disponível em: https://www.cloudkarafka.com/blog/part1-kafka-for-beginners-what-is-apache-kafka.html. Acesso em: 18 ago. 2023.

KOLB, Juliana, A estrutura do DAMA-DMBOK, 2020. Disponível em: https://jkolb.com.br/a-estrutura-do-dama-dmbok/. Acesso em: 18 ago. 2023.

KREPS, Jay, Questioning the Lambda Architecture ,2014. Disponível em: https://www.oreilly.com/radar/questioning-the-lambda-architecture/. Acesso em: 18 ago. 2023.

KUBERNETES, 2021. Disponível em: https://kubernetes.io/pt-br/docs/concepts/. Acesso em: 18 ago. 2023.

MELLA, Leoni. Docker e Docker Compose um guia para iniciantes, 2020. Disponível em: https://dev.to/ingresse/docker-e-docker-compose-um-guia-para-iniciantes-48k8. Acesso em: 18 ago. 2023.

MICROSOFT, 2023, O que é FinOps. Disponível em: https://learn.microsoft.com/pt-br/azure/cost-management-billing/finops/overview-finops. Acesso em: 18 ago. 2023.

MICROSOFT, Arquitetura de Microsserviços 2023. Disponível em: https://learn.microsoft.com/pt-

br/dotnet/architecture/microservices/architect-microservice-container-applications/microservices-architecture. Acesso em: 18 ago. 2023.

MICROSOFT, Arquiteturas de Big Data. Disponível em: https://learn.microsoft.com/pt-br/azure/architecture/data-guide/big-data/. Acesso em: 18 ago. 2023.

MURAKI, André, A revolução dos dados com Modern Data Warehouse, 2021.Disponível em: https://imasters.com.br/data/a-revolucao-dos-dados-com-modern-data-warehouse#:~:text=No%20ambiente%20virtual%2C%20o%20DW%20passou%20a%20ser,informa%C3%A7%C3%B5es%20armazenadas%20no%20final%20do%20processo%20de%20BI. Acesso em: 18 ago. 2023.

OBJECTIVE, O impacto de substituir arquiteturas monolíticas por microsserviços para os negócios, 2023. Disponível em: https://www.objective.com.br/insights/microsservicos/. Acesso em: 18 ago. 2023.

QNAP BRASIL. O que é virtualização, 2021. Disponível em: https://www.qnapbrasil.com.br/blog/post/o-que-e-virtualizacao. Acesso em: 18 ago. 2023.

RAJAGOPALAN, Rajesh Demystifying Data Mesh, 2021. Disponível em: https://www.peerislands.io/demystifying-data-mesh/. Acesso em: 18 ago. 2023.

REDHAT, 2018. O que é Virtualização. 2018. Disponível em: https://www.redhat.com/pt-br/topics/virtualization/what-is-virtualization. Acesso em: 18 ago. 2023.

REDHAT, Introdução ao DevOps, 2022. Disponível em: https://www.redhat.com/pt-br/topics/devops. Acesso em: 18 ago. 2023.

REDHAT, O que é arquitetura orientada a eventos?, 2019. Disponível em: https://www.redhat.com/pt-br/topics/integration/what-is-event-driven-architecture. Acesso em: 18 ago. 2023.

RODRIGUEZ, Leonardo, 2022. Easy Guide on Scraping LinkedIn With Python + Full Code!. Disponível em: https://www.scraperapi.com/blog/linkedin-scraper-python/. Acesso em: 18 ago. 2023.

S, Daiana, Microsserviços Python: Como construir aplicações distribuídas, 2023. Disponível em: https://www.homehost.com.br/blog/pythondjango/microsservicos-python/. Acesso em: 18 ago. 2023.

SHUKLA, Kapil, Build a data streaming pipeline using Kafka Streams and Quarkus, 2020. Disponível em: https://developers.redhat.com/blog/2020/09/28/build-a-data-streaming-pipeline-using-kafka-streams-and-quarkus. Acesso em: 18 ago. 2023.

SOLACE, The Complete Guide to Event-Driven Architecture, Disponível em: https://solace.com/what-is-event-driven-architecture/#:~:text=Companies%20of%20all%20sizes%20all%20over%20the%20world,4%20Federal%20Aviation%20Administration%205%20RBC%20Capital%20Markets. Acesso em: 18 ago. 2023.

SOLACE, The Complete Guide to Event-Driven Architecture. Disponível em: https://solace.com/what-is-event-driven-architecture/#:~:text=Companies%20of%20all%20sizes%20all%20over%20the%20world,4%20Federal%20Aviation%20Administration%205%20RBC%20Capital%20Markets. Acesso em 05 de agosto de 2023

TECNOLOGIA E NOTÍCIAS, AVIVATEC, Governança de dados: o que é e como aplicá-la corretamente. Disponível em: https://www.avivatec.com.br/governanca-de-

dados/#:~:text=A%20governan%C3%A7a%20de%20dados%20%C3%A9%20uma%20estrat%C3%A9gia%20baseada,passando%20pelo%20uso%2C%20at%C3%A9%20o%20descarte%20dos%20dados. Acesso em: 18 ago. 2023.

TRIGGO.AI. Princípios de DataOps. Disponível em:https://triggo.ai/blog/os-principios-de-dataops/. Acesso em: 18 ago. 2023.

TRUENET. O que é Virtualização?. 2021. Disponível em: https://blog.truenet.pt/virtualizacao/. Acesso em: 18 ago. 2023.

VASCONCELOS, L. C.; BRILHANTE, I.; LEMOS, S. Entenda como funciona streaming de dados em tempo real. Insight Lab, 2020. Disponível em: <https://www.insightlab.ufc.br/entenda-como-funciona-streaming-de-dados-em-tempo-real-2/>. Acesso em: 21 de set, 2023.

LEVY, E. Batch vs Stream vs Microbatch Processing: A Cheat Sheet. Upsolver, 2021. Disponível em: <https://www.upsolver.com/blog/batch-stream-a-cheat-sheet>. Acesso em: 21 de set, 2023.

PINHEIRO, M. Avro. Malum, 2019. Disponível em: <https://malum.com.br/wp/2019/07/09/avro/>. Acesso em: 21 de set, 2023.